SYSTEM AND METHOD OF DYNAMIC LOAD BALANCING ACROSS PROCESSOR NODES

TECHNICAL FIELD OF THE INVENTION

This invention relates to distributed processing, and more particularly, to a system and method of dynamic load balancing across processor nodes.

BACKGROUND OF THE INVENTION

In distributed processing architectures, multiple processing nodes share the work load according to some predefined load balancing algorithm. Conventional methods include round robin or weighted round robin, for example, which assign work to the processing nodes in a static or fixed manner. Furthermore, conventional methods do not fully and effectively utilize the additional processing power of improved processing nodes because the added capability or efficiency of these nodes are typically not taken into account in balancing the work load. For example, a system may include four processing nodes with different processing capacity, perhaps due to the different vintage of the processors, with the new processors having improved capacity. Static load balancing methods do not assign more work to those processor nodes with higher capacity to take advantage of the added computing power. To fully exploit the continuous increases in processing power of newer computer processor designs, work load balancing should allow processor cluster expansions and upgrades with higher capacity processor nodes while not requirement replacement or retirement of existing older processing nodes.

SUMMARY OF THE INVENTION

Therefore, it is desirable to provide a dynamic load balancing methodology which assigns work to multiple processing nodes so that the nodes function at an approximately equal percentage of each node's full processing capacity in order to fully take advantage of the processing capacities of the processing nodes.

In accordance with an embodiment of the present invention, a method of dynamically balancing work among a plurality of processing nodes is provided. The method includes the steps of periodically updating a node occupancy value at each of the plurality of processing nodes, communicating the respective node occupancy value of each processing node to at least one work originator node, storing the node occupancy values of the plurality of processing nodes at the at least one work originator node, and selecting, by the at least one work originator node, a processing node to perform a particular task in response to the node occupancy values of the processing nodes.

In accordance with another embodiment of the present invention, a method of dynamically balancing call processing tasks among a plurality of call processing nodes in a telecommunications switch is provided. The method includes the steps of periodically updating a node occupancy value at each of the plurality of call processing nodes, communicating the respective node occupancy value of each call processing node to at least one work originator node operable to receive incoming calls, storing the node occupancy values of the plurality of call processing nodes at the at least one work originator node, and selecting, by the at least one work originator node, a call processing node to process the incoming call in response to the node occupancy values of the call processing nodes.

In accordance with yet another embodiment of the present invention, a telecommunications system is provided. The telecommunications system includes a plurality of call processing nodes and at least one incoming call receiving node. The plurality of call processing nodes each periodically calculates and updates a respective node occupancy value, and communicates the respective node occupancy value to at least one incoming call receiving node. The at least one incoming call receiving node stores the node occupancy values of the plurality of call processing nodes, and selects

a call processing node to process the incoming call in response to the stored node occupancy values of the call processing nodes.

The present invention thus dynamically balances the processing load of the processing nodes as a percentage or relative to the total capacity. As a result, the work load can be more evenly and more intelligently distributed to fully take advantage the higher capacity of newer and faster computer processing technology. Because the node occupancy information is communicated in the message header of existing message traffic, little or no overhead is expended to accomplish this task. The use of an open loop feedback design versus a closed loop feedback design provides a more flexible load balancing scheme. In addition, each node in the system is able to calculate its own occupancy rate in the manner best suited to that particular node or best for overall system performance.

BRIEF DESCRIPTION OF THE DRAWINGS

For a more complete understanding of the present invention, the objects and advantages thereof, reference is now made to the following descriptions taken in connection with the accompanying drawings in which:

FIGURE 1 is a simplified block diagram of a distributed processing system containing work originators and work performers;

FIGURE 2 is a simplified block diagram of a distributed processing system set in a telecommunications environment;

FIGURE 3 is a flowchart of node occupancy value calculation according to an embodiment of the teachings of the present invention;

FIGURE 4 is a flowchart of sending an inter-node message containing a node occupancy value according to an embodiment of the teachings of the present invention;

FIGURE 5 is a flowchart of receiving an inter-node message containing a node occupancy value according to an embodiment of the teachings of the present invention; and

FIGURE 6 is a flowchart of selecting a processing node according to the node occupancy values according to an embodiment of the teachings of the present invention.

DETAILED DESCRIPTION OF THE DRAWINGS

The preferred embodiment of the present invention and its advantages are best understood by referring to FIGURES 1 through 6 of the drawings, like numerals being used for like and corresponding parts of the various drawings.

FIGURE 1 is a simplified block diagram of a distributed processing system containing work originators 10 and work performers 12. Work originators 10 and work performers 12 may be computing platforms, processor-based devices, or any equipment or processes that are capable of carrying out a logical sequence of steps. Work originators 10 may receive requests for work from other devices (not shown) via networks and are operable to assign work to one or more work performers 12. Work performers 12 operate in a load-sharing manner so that any one work performer 12 does not become overwhelmed with work.

In systems that have processing nodes that have dis-similar processing capacities, traditional static load balancing methods do not take full advantage of the higher processing power of the processors. Typically, the processing capacity of processing nodes are used to execute operating system and application support software, standby applications, active applications, and load-shared applications. The amount of processing capacity used for these applications typically vary among the processing nodes because different set of applications may run on different processing nodes, which may also change over time. The processing capacities used for the applications are also different due to the different processing power of the nodes. Therefore, assigning a discrete unit of work to one processing node may cause its work load to change by X%, while the same unit of work may cause another processing node to change its work load by Y%. Therefore, load balancing methods that rely on units of work or a round robin scheme do not fully exploit the higher processing power of newer processing nodes.

In an embodiment of the present invention, the work load is shared among work performers 12 so that each work performer 12 functions at more or less an equal percentage of its own full capacity. Work originators 10 and work performers 12 communicate by inter-node messaging. In the present invention, the status of each work performer's work load is inserted into each inter-node message originating from that work performer destined for a work originator. Each work originator 10

maintains a record of all work performer's current load condition and consults this record whenever work needs to be assigned to a work performer. On the basis of the work load record, a work performer is selected and assigned the new work. This load balancing scheme is an open loop feedback design that dynamically assigns work based on the percentage of capacity available to do the work at each work performer. Details of the invention are described below.

FIGURE 2 is a simplified block diagram of a distributed processing system 20 set in a telecommunications environment. In particular, system 20 is an integrated media switching platform. System 20 includes work originators 10, which are multi-service fabric (MSF) 24 and signaling gateways (SGW) 26. System 20 also includes work performers 12, which are multi-service controllers (MSC) 30 coupled to work originators 10 via networks, network servers and/or network switches 28. Network servers 28 may be Ethernet switches, for example. Work originators 10 interface with public switched telephone network (PSTN) 32, asynchronous transfer mode (ATM) network 34, customer premises equipment (CPE) such as a private branch exchange (PBX) 36, integrated digital loop carrier (IDLC) 38, simple network management protocol (SNMP) network management system (NMS) 40, user interface (I/F) 42, and other network nodes. As voice or data calls are being originated, work originators 10 select a work performer 12 to handle the call based on its current work load. The work load is distributed so that the work performers all perform at substantially the same percentage of each processor's full capacity.

FIGURE 3 is a flowchart of node occupancy value calculation according to an embodiment of the teachings of the present invention. Each work performer is operable to calculate or otherwise determine its own occupancy rate or value. In one embodiment, the node occupancy value is determined on a periodic basis, as shown in block 60. If it is time to calculate or determine the current occupancy value, then it is calculated according to a predefined method or formula, as shown in block 62. A combination of percentage of processor occupancy and the length of the incoming work queue are factors that can be used to calculate the occupancy value. For example, a processing node may calculate its node occupancy value by:

$$Processor\_Occupancy\% * n + Pending\ Queue\_Length * m,$$

where n and m are tuning factors; for example, n=.8 and m=1 in one embodiment. On the other hand, another processing node may be instructed to output a high occupancy value rather than to make a true determination in order to keep out new work because it is currently testing a new software load, for example. All processing nodes in a system may use the same calculation method, or different processing nodes may use different methods in the manner best suited to each individual node. As a further example, the occupancy value calculation may provide hysteresis to smooth the resultant output to avoid large swings in the node occupancy value. The newly determined node occupancy value is then stored or used to update a known memory location, as shown in block 64.

FIGURE 4 is a flowchart of sending an inter-node message containing a node occupancy value according to an embodiment of the teachings of the present invention. A work performer, during the normal course of events, communicates with work originators by sending inter-node messages. For every message a work performer sends, it sends a status of its current work load. In blocks 70 and 72, the work performer reads the current node occupancy value and insert the value into a appropriate predetermined location or field in the message header of an inter-node message. Also included in the header is sender ID or address and recipient ID or address. The message is then sent to the destination, as shown in block 74. In this manner, the recipient of the message is provided a current status of the work load of the sender work performer.

FIGURE 5 is a flowchart of receiving an inter-node message containing a node occupancy value according to an embodiment of the teachings of the present invention. A work originator receives an inter-node message from another node or a work performer, as shown in block 80. The work originator extracts the node occupancy value from the predefined field in the message header as well as the sender node's unique ID or address, as shown in block 82. The node occupancy value is then stored in a node occupancy table indexable by the node IDs, as shown in block 84. The node occupancy table is stored in the respective memory of each work originator node.

FIGURE 6 is a flowchart of selecting a processing node according to the node occupancy values according to an embodiment of the teachings of the present invention. A work originator receives work from an external source or another node in the network, as shown in block 90. For example, work may be in the form of signaling data and voice data for a telephony call received by multi-service fabric 24 and signaling gateway 26. As part of the process of selecting a work performer to handle the work, the work originator reads the node occupancy table to determine which work performer(s) is(are) capable of handling the work and is(are) among the lowest in terms of occupancy status, and sends the work to the selected node(s), as shown in blocks 92 and 94. In one exemplary embodiment, the work originator may randomly select a node from the lowest occupied third of the available processing nodes. The work performer then sends a request to the selected work performer to perform the task. For example, multi-service fabric 24 prepares and sends a call setup message to the selected work performer so that it may handle the incoming call.

When the work performer is chosen in this manner, the dynamic work processing load for each work performer as a percentage or relative to the total capacity is taken into account. As a result, the work load can be more evenly and more intelligently distributed to fully take advantage the higher capacity of newer and faster work performers. Because the node occupancy information is communicated in the message header of existing message traffic, little or no overhead is expended to accomplish this task. The use of an open loop feedback design versus a closed loop feedback design provides a more flexible load balancing scheme. Each node in the system is capable of calculating its own occupancy rate in the manner best suited to that node or best for overall system performance.

While the invention has been particularly shown and described by the foregoing detailed description, it will be understood by those skilled in the art that various changes, alterations, modifications, mutations and derivations in form and detail may be made without departing from the spirit and scope of the invention.